International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 12, No.1 (Dec-2022)

https://dx.doi.org/10.12785/ijcds/1201112

Support Vector Machine based multi-View hashing approach for Near Duplicate Video Retrieval

Dhanashree Phalke[1] **and Sunita Jahirabadkar**[2]

[1]*Research Scholar, Department of Technology, Savitribai Phule Pune University, Pune, India*
[2]*Cummins College of Engineering for Women, Pune, India*

Received 19 Jan. 2022, Revised 7 Sep. 2022, Accepted 1 Dec. 2022, Published 8 Dec. 2022

Abstract: Near duplicate videos (NDVs) are the primary concern for cloud storage and web search. Similar video-sharing triggers issues related to copyright and financial loss to the maker of the video. We propose a near-duplicate video retrieval (NDVR) system. The proposed algorithm is trained and tested on the benchmark standard dataset CC-WEB-VIDEO. For video processing, we first split it into frames. We have processed 48 GB of frames retrieved from 80GB of video dataset. Hue Saturation Value (HSV) and Local Binary Pattern (LBP) are used to capture the global and local features of frames. It is observed that 3% to 10% of frames have similar frames. Therefore, the kernel-based component analysis (KPCA) algorithm is used to reduce the redundant frames. A deep Convolution Network-based VGG16 algorithm is also used to identify the best strategy for NDV. Finally, the feature extracted from all three techniques, HSV-KPCA, LBP-KPCA, and VGG16, are trained and tested on a radial basis function-based support vector machine (RBF-SVM) classifier. RBF is used to address the non-linearity of nonredundant frames. Results are compared with state-of-the-art algorithms for NDVR. The proposed system reports a higher Mean Average Precision (MAP), Area under the curve (AUC), and accuracy than previous NDVR systems.

Keywords: HSV, LBP, VGG16, Near Duplicate Video retrieval

1. Introduction

The last two decades are also known for the online era- which has witnessed a drastic volume rise in media content, especially in video-related applications on social media. Video-sharing and broadcasting have become common. Searching for exact content or near-duplicate content from the web application is a concern. Internet is flooded with duplicate or near-duplicate video (NDV) through editing and distribution services [1]. To remain in the race to become number 1 in the video application, the company requires maintaining information of duplicate or near-duplicate video of query videos. NDVs can impact a large scale on video-related applications; hence, near-duplicate video retrieval (NDVR) is the concept that is largely in use. NDVR reduces the cost of space for storing unique content and helps in searching for similar content or near-duplicate videos. NDVR will also help in copyright protection by identifying the near-duplicate video resulting from compromising real video through the editing and redistribution process. NDVR is a process of mapping and searching for similar videos from a set of existing videos [2]. It consists of three main phases; a) Frame extraction, b) Feature extraction and reduction, and c) classification.

From a video, keyframes are extracted at regular intervals. There are always possibilities of getting similar types of frames from the given video. Hence, Multiview hashing [3], [4] maps all the keyframes into a hash. The hash bit will eliminate the duplicate frame as they map to the same hash. Feature extraction techniques such as HSV [5], [6] , LBP [7], [8], [9] and VGG16 [10] are used on the keyframe to retrieve real values from each frame using image analysis. The extracted information increases the effectiveness of the system. Similar video can be retrieved by analyzing and classifying all the extracted features of the video's frames, for which binary classification is used for dual sampling using SVM [11], [12], [13], [14].

2. Materials and Methods

In this paper, we have worked on the features extracted from HSV, LBP, and VGG16, and finally compared the accuracy through binary classifier SVM, as each query video can be classified into {near duplicate video (NDV), Non-NDV}. Open database CC-WEB-VIDEO is used to train and test our model, examine and compare the proposed system with other NDVR systems. Comparing parameters Precision, Recall, and MAP are used to compare the model's accuracy.

E-mail address: daphalke@dypcoeakurdi.ac.in, sunita.jahirabadkar@cumminscollege.in

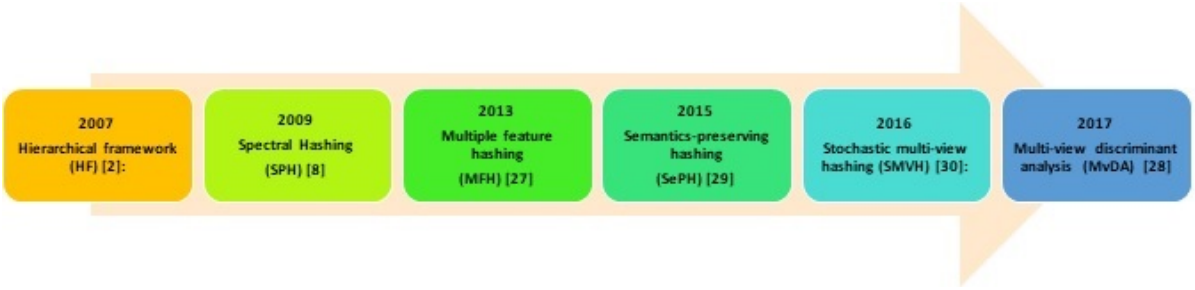

Figure 1. NDVR Time Frame Research

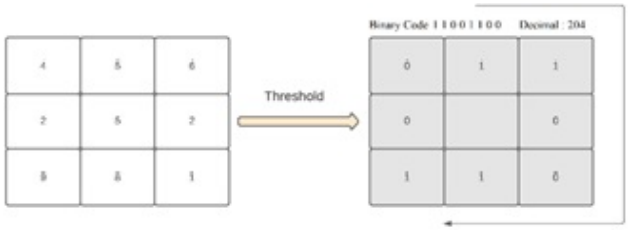

Figure 2. A basic LBP operation example

The rest of the paper is prepared; materials and methods are discussed in section 2, followed by the Proposed Model Architecture in section 3. section 4 deals with the dataset and detailed experiment analysis. Finally, the paper is concluded in section 5.

Figure 3. Local Binary Pattern (LBP) on Frame 1 of video "The Lion Sleeps Tonight."

Figure 1 shows important development in the field of video retrieval and classification. In 2007, Hierarchical Framework [HF] used the first global feature model (HSV). Subsequently, in 2009, (PCA-SIFT) [15] based local features were used to classify query videos. Spectral hashing (SPH) [8] uses HSV, LBP features, and Hash codes are evaluated using eigen-vector or Laplace graph. In 2015, Semantics-preserving hashing (SePH) [16] used probability distribution with hamming distance-based hash codes through divergence algorithms on supervised data. Multiple feature hashing (MFH) [17] learns the hash function by considering the graph model that preserves HSV and LBP features. Later in 2017, multi-view-based methods that work with the local and global view and variations among samples have reported the best MAP and accuracy algorithms; (MvDA) [18] known as multi-view-discriminant-analysis. Another model, Stochastic Multi-view Hashing (SMVH) [19], learns hash codes by considering different features using a probabilistic model. We use the MVH methods to map hash codes and kPCA to reduce the dimension of components extracted from MVH and LBP.

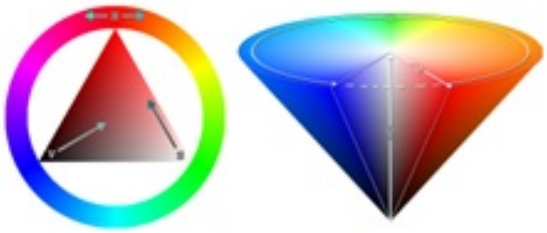

Figure 4. HSV Color Model [20]

Considerable effort has been invested by the research community on the problem of NDVR. However, many state-of-the-art methods adopt hashing and discriminant analysis approach for both development and evaluation. For instance, some methods of hashing functions are based on sample frames from the evaluation dataset, and as a result, their reported retrieval performance is often exaggerated. This leads to specialized solutions that typically exhibit poor performance when used (without tuning) on the different video sets. For instance, each video has multiple frames and has different pixels intensity. We believe Frame global features and local features can contribute better and more useful features for classifications. Also, we are motivated by the performance of deep learning to address the multimedia problem. Hence, using the convolution model can be a better approach for extracting full, meaningful features.

Figure 5. HSV on Frame 1 of video "The Lion Sleeps Tonight."

A. *Problem Formulation*

A framework comprising three stages of matching is developed to solve the problem of near-video duplicate identification. Let $V = \{v_1, v_2, \ldots, v_k\}$ be a category of k number of videos and q be the query video to be searched in space V for potential near-duplicate videos. Video v_i,

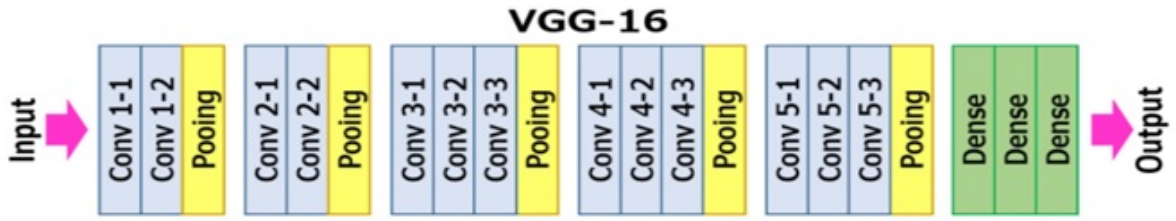

Figure 6. Architecture of VGG16 [21]

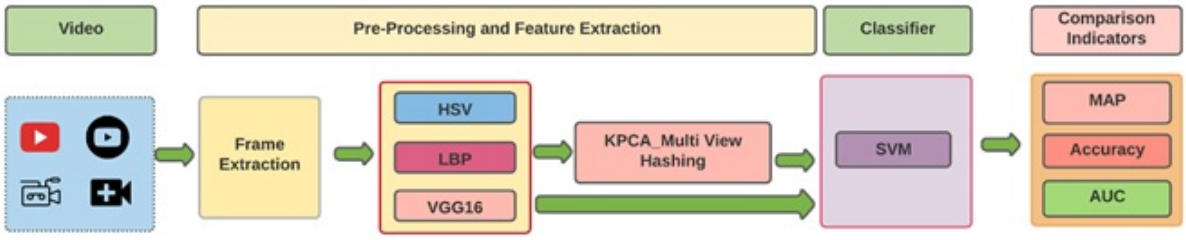

Figure 7. Proposed Architecture for NDVR

where $i = 1, 2 \ldots, k$ is split into n frames in time interval r, such that $F_i = \{f_{i1}, f_{i2}, \ldots f_{in}\}$, where f_{ij} denotes the j^{th} frame of i^{th} video v_i. Their exit many feature extraction methods [22], [23], [15], [24] to retrieve numerical characters of the keyframes F_i, but we targeted the research problem in a general manner. Thus, our experiment considers the three most used feature extraction algorithms, HSV, LBP [25], and VGG16. HSV extracts global features, and LBP retrieves the local texture of frames, whereas VGG16 uses deep learning convolution techniques to extract meaningful features. This results in three separate feature vectors for each of the n keyframes, stored in 2D feature matrix $X_M = [x_{ij}^M], X_L = [x_{ij}^L]$ and $X_V = [x_{ij}^V]$ with the size of np_1 , np_2 and np_3 respectively. The three-column vector $x_i^M = [x_{i1}^M, x_{i2}^M, \ldots, x_{ip1}^M], x_i^L = [x_{i1}^L, x_{i1}^L, \ldots, x_{ip2}^L]$, and $x_i^V = [x_{i1}^V, x_{i2}^V, \ldots, x_{ip3}^V]$ are used to denote the H SV, LBP, and VGG16 features for the i^{th} keyframe.

B. Local Binary Pattern (LBP)

LBP is mainly used to summarize the local structure of input images. Recent development and application of LBP are primarily found in computer vision and image processing. LBP model first identifies the binary code of structure by comparing the neighbor cell with the center cell position and then labels its corresponding decimal value as histogram output.

Its steps are shown in Figure 2, 8 pixels are considered, and each pixel is compared with center pixels over a $3x3$ neighbor pixel matrix. The resultant matrix is encoded in 0 and 1, considering value comparison. If the pixel value is greater than or equal to the center pixel, the corresponding cell is 0; otherwise, 1. Now clockwise direction movement provides us binary values. The corresponding binary value decimal number is used to label the pixel.

LBP codes have a significant limitation: a small $3x3$ neighborhood of radius 1 (distance of comparison pixel from center cell) cannot detect essential features on a large dataset [26]. Also, working with a smaller radius will increase the computational cost of the entire model. Hence [27] generalize the concept of using different radius.

Figure 3 shows one example of LBP taken from the movie's first frame, "The Lion Sleeps Tonight". For a Q Sampling point on a circumference of Radius R, LBP can be calculated using equation (1).

$$L_{Q,R}(X_C, Y_C) = \sum_{Q=0}^{Q-1} D(i_Q - i_C)2^q \tag{1}$$

where i_Q and i_C are, a gray level value of center and surrounding pixel respectively and function D(x) is defined as

$$D(x) = \begin{cases} 1 & x \geq 0 \quad (2) \\ 0 & x < 0 \quad (3) \end{cases}$$

This preserves pixels' intensity and local information in the monotonic gray-scale transformation.

C. Hue Saturation Value (HSV)

The HSV scale provides a numerical reading of the input image corresponding to different color name in it, shown in figure 4. Hue is valued between 0 to 360^o. Cyan and magenta found in range $[181^o, 240^o]$ and $[301^o, 360^o]$ degrees, respectively, whereas the filling space is analyzed from 0 to 100%. HSV color models are beneficial for choosing accurate art colors, color swatches, and digital

TABLE I. CC-WEB-VIDEO Data Set Description

Video Type	Total Video	No. of Frames in Interval r	Ground Truth		70% Training Data Set		30 % Testing Data Set	
			NDVR	Non-NDVR	Train-NDVR	Train-Non-NDVR	Test-NDVR	Test-Non-NDVR
VT1	812	100298	341	471	234	334	107	137
VT2	594	171586	124	470	88	85	36	39
VT3	440	22961	201	239	132	176	69	63
VT4	353	19062	162	191	110	137	52	54
VT5	404	113544	94	310	66	65	28	29
VT6	781	167673	46	735	32	32	14	14
VT7	365	50215	159	206	110	145	49	61
VT8	552	140592	45	507	31	32	14	13
VT9	238	48006	26	212	18	18	8	8
VT10	305	78751	54	251	36	39	18	15
VT11	381	74644	69	312	44	52	25	17
VT12	897	151615	171	726	125	114	46	57
VT13	419	33644	387	32	270	23	117	9
VT14	108	22677	77	31	51	24	26	7
VT15	1791	392920	713	1078	512	741	201	337
VT16	208	47494	20	188	14	14	6	6
VT17	657	141918	208	449	142	317	66	132
VT18	501	95242	129	372	93	87	36	42
VT19	570	68073	237	333	164	235	73	98
VT20	198	60432	88	110	62	76	26	34
VT21	458	79291	56	402	35	43	21	13
VT22	428	80243	84	344	59	58	25	26
VT23	1352	295718	272	1080	193	187	79	85
VT24	313	43414	26	287	18	18	8	8

images. This paper converts video frames from an RGB color format to HSV's proposed color areas, HSV uses eq (4) - eq (6).

Each pixel of the video frame is composed of three component red (R), green (G), and blue (B). HSV is the transformation of RGB color space. In order to derive chroma $C = M - m$, where $M = max(R, G, B)$ and $m = min(R, G, B)$. Now the Hue, saturation, and value can be derived using eq (4) to eq (6)

$$h = \begin{cases} 0 & if, M = m \quad (4a) \\ (60^0 x\left(\frac{G-B}{C}\right) + 360^0) mode 360^0 & if, M = r \quad (4b) \\ 60^0 x\left(\frac{B-R}{C}\right) + 120^0 & if, M = g \quad (4c) \\ 60^0 x\left(\frac{R-G}{C}\right) + 240^0 & if, M = b \quad (4d) \end{cases}$$

$$S = \begin{cases} 0 & if, M = 0 \quad (5a) \\ \left(\frac{C}{M}\right) = 1 - \frac{m}{M} & Otherwise \quad (5b) \end{cases}$$

$$v = M \quad (6)$$

where red (r), green (g) and blue (b) normalized in range [0, 1]. And according to [28] quantization of h can be given by:

$$\overline{h} = \begin{cases} 0 & if, h \in [0, 20) \cup (315, 360) \quad (7a) \\ 1 & if, h \in [20, 50) \quad (7b) \\ 2 & if, h \in [50, 75) \quad (7c) \\ 3 & if, h \in [75, 155) \quad (7d) \\ 4 & if, h \in [155, 195) \quad (7e) \\ 5 & if, h \in [195, 275) \quad (7f) \\ 6 & if, h \in [275, 315) \quad (7g) \end{cases}$$

After completing the hue component, the s and v planes are normalized in the range [0, 255]. Figure 5 shows the transformation of the first frame of the video title "The Lion Sleeps Tonight" into the HSV image. For high transformation, high H:179,high S:255, high V: 213, whereas Lower H, Lower S, and Lower V is all set to 0.

D. Multi-View Hashing

Multi-View Hashing (MVH) is composed of k hash function $\{H_1(.), H_2(.), \ldots, H_k(.)\}$. Each hash function $H_i(.)$, translate i^{th} feature of the frame into a binary code of length k for each keyframe. The n keyframes, will have derived strings of size nk binary hash code matrix $H = [h_{ij}]$. Now, for n frames, $|H| = nxs$ binary string. Now, the similar

TABLE II. HSV Values to extract HSV features from query video frames of each 24-video set

Video Type	High H	High S	High V	Low H	Low S	Low V
VT1	179	255	213	0	0	0
VT2	179	181	255	0	83	139
VT3	176	186	160	46	0	0
VT4	138	183	231	11	19	0
VT5	179	255	255	49	95	3
VT6	178	185	255	0	0	145
VT7	178	255	208	0	108	31
VT8	147	74	255	33	0	0
VT9	13	255	255	0	142	114
VT10	179	255	255	1	0	0
VT11	179	255	255	37	55	39
VT12	179	169	255	25	0	34
VT13	179	246	144	0	50	0
VT14	161	242	255	23	103	0
VT15	82	205	255	0	99	58
VT16	76	114	255	5	0	91
VT17	179	255	255	0	103	80
VT18	179	70	255	142	9	86
VT19	20	255	255	0	50	120
VT20	33	255	239	0	111	0
VT21	20	194	194	0	17	138
VT22	137	233	249	23	81	28
VT23	12	255	111	3	64	45
VT24	87	59	168	74	9	0

frames will have identical hash codes. Thus, referring to unique binary hash codes can eliminate the duplicate frame and reduce the space and time complexity of the model. MVH extracts u individual frames such as $|u| < |n|$, which has a unique hash bit and can be extracted using eq (8).

$$\overline{F_i} = M(F_i) \tag{8}$$

where, $\overline{F_i} = \{f_{i1}, f_{i2}, \ldots, f_{iu}\}$, is the set of unique frames such that $|\overline{F_i}| \leq |F_i|$. $M(.)$ is the function that computes MVH on the frames of F_i.

TABLE III. MAP comparison of the dataset

Systems	Dataset:CC-WEB-VIDEO
HF [2]	0.952
SePH [8]	0.8837
SPH [16]	0.848
MFH [17]	0.934
SMVH [18]	0.957
MvDA [19]	0.971
MVH-HSV-SVM	0.978
MVH-LBP-SVM	0.971
MVH-HSV-LBP-SVM	0.9787
VGG16-SVM	0.9937

All the overlapped frames will have the same hashing, reducing the system's time and space complexity by saving unnecessary computation on similar frames. Working on a no-redundant edge guarantee that the future feature extraction will have meaning and essential features component for machine learning, Eq. (2), produces the 2-D matrix derived from method $FeatureExtractionF(.)$;

$$S = \mathcal{F}(\overline{F_i}) \tag{9}$$

S is a two-dimensional matrix of size up, where u is the number of unique frames, and p is each frame's extracted feature by method $F(.)$. F_i results in the set of unique frames $\overline{F_i} = \{f_{i1}, f_{i2}, \ldots, f_{iu}\}$. As $|\overline{F_i}| \leq |F_i|$, will have a remarkable impact on time and space complexity by saving unnecessary computation on similar frames. For better machine learning, feature extraction is the initial and essential stage. This paper implements and compare three HSV, LBP, and VGG16 as feature extraction methods.

E. VGG16

VGG16 is a deep learning algorithm based on a convolutional neural network model (CNN). It uses smaller filters. The VGG16 is deeper in comparison with CNN models. Figure 6 depicts the flow of the VGG16 model. Each frame is considered a color image of tri-dimension (224, 224, 3). The initial two layers of VGG16 have 64 filters of $3X3$ size channels with similar padding. Max pool layer of stride $(2, 2)$, again 256 filter of Convolution layers of size $3x3$ is

applied, followed by the stride of $(2, 2)$ in the max-pooling layer as per the previous layer. The same is repeated of two 256 convolutions of filter size $3x3$ is applied. There are 2 sets of 3 convolution layers, and one max-pooling layer is applied, where each layer has 512 of size $3x3$ filters with similar padding. Image is now passed through 2 convolution layers and padding of 1-pixels to prevent the spatial feature of the image. After executing the stack of convolution layers and max-polling layers, we get $(7, 7, 512)$ feature vectors. At this stage, flatten of the feature vector is made of numerical (1,25088) vector length. This numerical vector is retrieved and stored for the training and testing of the classifier. It is prolonged to train the VGG16 model on GPU for 18 days to process altogether 60 GB video data set using *"Keras package"* on python environment. The size of VGG-16 trained weights is 528 MB. So, it takes a lot of disk space and bandwidth, making it inefficient considering space and time complexity.

3. Proposed Model Architecture

The proposed architecture comprises of two crucial stages: Extracting meaningful features and identifying similar videos based on query video. Feature extraction is the process of identifying the numerical characteristics of video contents. The basic framework of the Proposed model system is illustrated in Figure 7. It comprises three phases: preprocessing, Feature Extraction, and Classification training and testing.

A. Phase1:

Preprocessing- Video V_i is split into a different frame f_{ij} where, $j \in 1, \ldots, n$ in and interval of r, such that $n = \frac{|V_i|}{r}$, where $|V_i|$ is the duration of i^{th} video. Followed by feature extraction disused in Section 2, it is used to extract keyframe F_i Key information. Each frame is treated as an image that has a three-pixel value RGB. For j^{th} keyframe of video V_i, we get a 2-D feature matrix X_M,X_L and X_V of different vector lengths for HSV, LBP, and VGG16 features. For the image of size $224x224$, HSV, LBP and VGG16 extracted features of dimensions 240, 59, and 25088. Now, Multi-View Hashing is used to extract the binary pattern of each feature frame. All duplicate frames will have the same binary pattern; thus, considering one copy of the structure from each binary code will result in a unique frame list $\overline{F}$. It is observed that each video has 3% to 10% duplicate frames considering short boundary, visual information, and movement analysis of the object. We have also used the KPCA algorithm for feature reduction. The reduced set of unique frames will fasten the computation process.

B. Phase2:

The paper's objective is to compare the feature extraction algorithm concerning the accuracy, MAP, and AUC. Support vector machine with kernel function is used to classify extracted features. The major strengths of SVM are that it does not stick in local minimal, and easy to train and test and has got very least parameter to tune. However, sometimes, it is hard to classify nonlinear data items. Kernel functions are mostly used to map nonlinear data into a higher plane for smooth linear recognition by SVM. Looking into the discontinuities among the data set, we have used the exponential radial basis function [29], [17], [18], [16], [19]. The choice of appropriate kernel function can differ concerning the choice of the data set.

The data sets with various combination of parameters $\{C, \gamma\}$ has been implemented, in which the parameter C is chosen from $\{2-5, 2-4, \ldots, 25\}$ and γ from $\{2-15, 2-14, \ldots, 2-1\}$. Features are split into 70-30 ratios as training and testing data. SVM is trained and tested over query video, confusion matrix, AUC parameter precision, and recall is evaluated, and AUC is compared for all three sets of features X_M, X_L, and X_V. The RBF kernels are the most general and broadly used kernels compared to the Gaussian distribution. The RBF kernel function for two feature vectors can be X and Y can be represented as $k(X, Y) = exp\left(\frac{\|X-Y\|^2}{2\sigma^2}\right)$, where σ is the variance, and $\|X - Y\|^2$ represents the squared euclidean distance between two feature vectors.

4. Experimental Analysis

The experiment was run on Python 3.9 compiler over Spyder Open-Source tools at 64-bit window 10, 12 GB RAM, Intel Core i3 processor with 2.66 GHz processor platform.

A. Dataset

The proposed model has been trained and tested on CC-WEB-VIDEO [30]. Total number of videos available in the dataset is 12,790. All the videos are collected from Google, Yahoo, You Tube. It composes of 24 different sets of videos. Details are discussed in figure **??** . On interval r = 20, we get 398,015 number of keyframes. We retrieve 24 different set of query videos using the ground truth of video information. The dataset creator has reported a manual label process to describe each video. There exist 7 class labels in the ground truth. Class labels are $\{X, E, S, L, V, M\}$, where X represents not similar video, E is exact, and $\{S, L, V, M\}$ are similar videos. The entire dataset can be divided into two class labels Near Duplicate Video as a video with labels E, S, L, V, and M, and Non-Duplicate video with class label X.

B. Result Analysis

From CC-WEB-VIDEO datasets, keyframes are extracted on interval r = 20 for each video. MVH algorithm is used to extract hash map binary code for each image. By considering uniqueness as the key, we observe 3 to 10% similarity in the frames. Image features are extracted using state-of-the-art HSV, LBP, and VGG16 [31]. HSV extracts 162 feature vectors, and LBP produces 256 feature vectors as a mean of feature vector of each image. Table II shows the value used in our paper to extract HSV of 24 video types of query images belongs to the CC-WEB-VIDEO dataset. LBP, histogram of length 256 is achieved by considering *radius* = 8. Using [29], we further reduce the length from

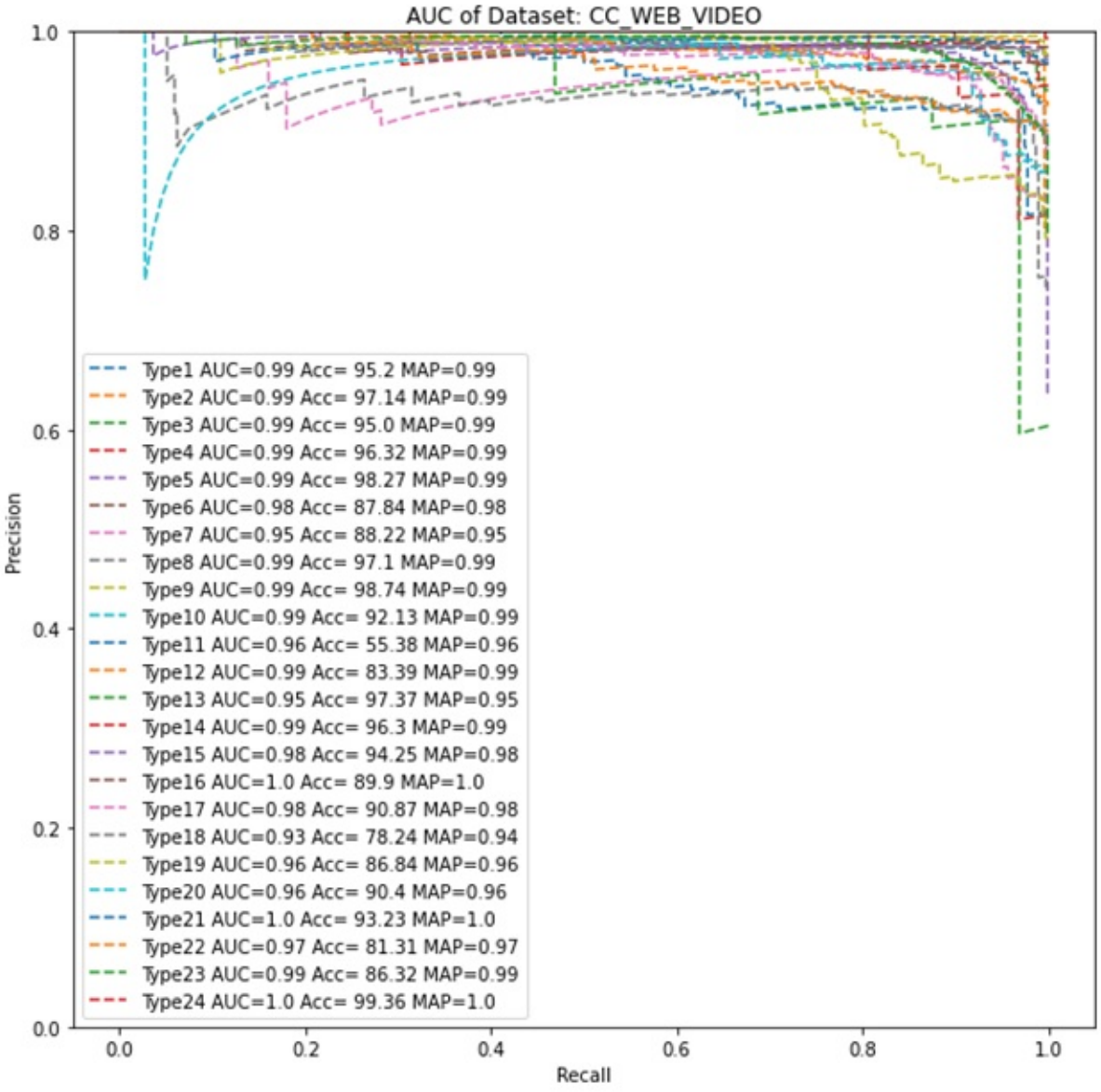

Figure 8. AUC, MAP, and Accuracy comparison using MVH-HSV algorithm

256 to 59. Whereas a total of 25088 features per image are extracted using VGG16.

The proposed experiment is carried over the CC-WEB-VIDEO dataset, where the training and testing sets are randomly split in a ratio 70:30 percent, respectively. The extracted features possess low computational complexity than [17], [18]. The performance of the proposed system is measured through three measuring indicators, Mean Average Precision (MAP), Precision-Recall ROC Curve, and Area Under Curve (AUC). MAP compares the ground truth with the predicted label and returns a score. Max scores depict better accuracy of the model. Precision, recall, and other effective indicators can be derived using the following equation:

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

The recall is also referred to as sensitivity, and precision is a positive predictive value. Where TP is a true positive that correctly indicates the presence of the condition, FP is a false positive, a result that wrongly means the particular conditions are present, and FN is a false negative, a result

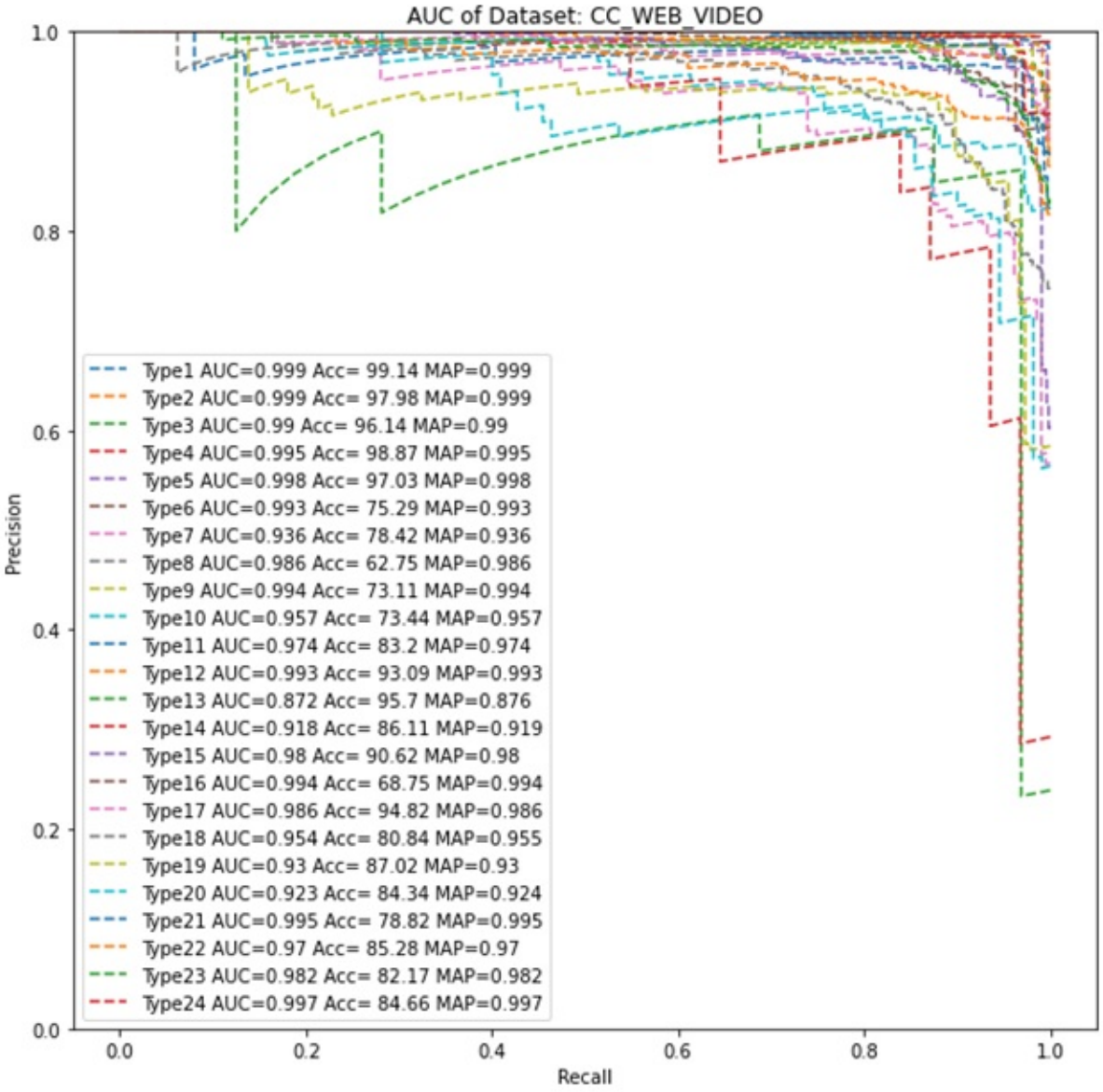

Figure 9. AUC, MAP, and Accuracy comparison using MVH-LBP algorithm

that wrongly indicates the particular conditions is absent. *MAP* for a set of queries can be derived using the equation:

$$MAP = \frac{\sum_{v=1}^{N} AveP(q)}{N} \tag{12}$$

Where q is the number of query videos. And $AveP(.)$ can be computed as

$$AveP = \sum_{z=1}^{N} P(z)\Delta r(z) \tag{13}$$

Where z denotes the rank of the video sequence, N denotes the total retrieved video, $P(z)$ is the precision, and $r(z)$ is the change in the recall.

Table III shows the performance comparison using MAP of the proposed architecture with methods disused in literature. In CC-WEB-VIDEO, the average MAP reported for all 24 video types using the MVH-HSV algorithm is around 0.9367, out of which 11 video types have been reported MAP > 0.95, 7 video types reported MAP > 0.90 remaining 6 Video Type has MAP > 0.85. In MVH-LBP, 17 Video set MAP > 0.95, 4 video type MAP > 0.9, 1 video type MAP > 0.8 and 0.75 each, and 1 video type particular set 12 is MAP reported to 0.4, thus dropping the

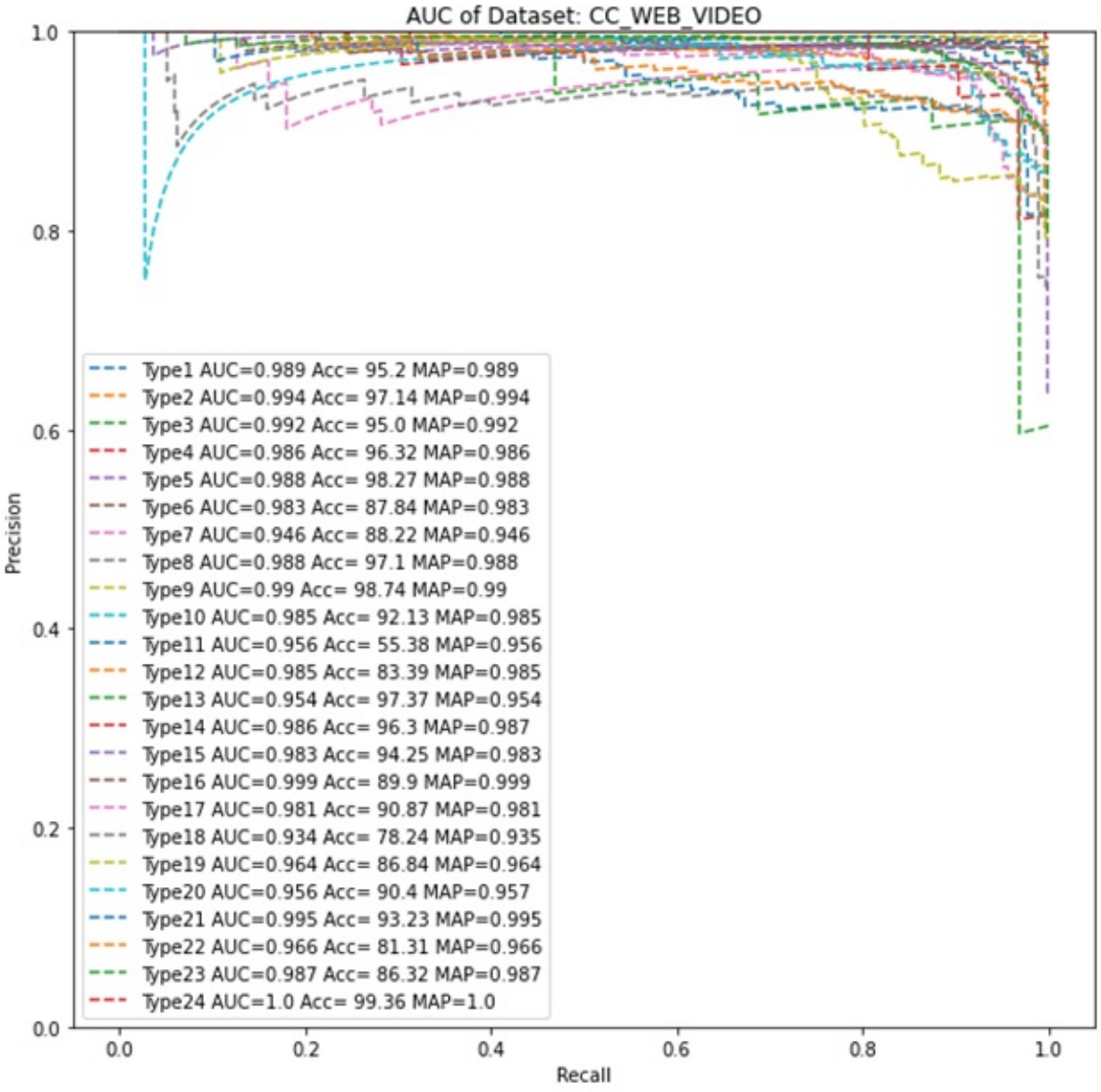

Figure 10. AUC, MAP, and Accuracy comparison using MVH-HSV-LBP Algorithm

overall MAP average to 0.938. The proposed approach has shown significantly better performance in comparison with HF [2], SePH [8], SPH [16], MFH [17], and SMVH [18]. The performance of LBP and HSV is slightly lower when compared independently with [19], but the joined effect of the feature of MVH-LBP-HSV better performs MvDA [19]. MvDA depends on global structure, but utilizing the low and high structure of LBP and HSV together has better MAP. MVH-LBP-HSV 18 video set reports MAP > 0.95, and 6 video set reports MAP > 0.90. An experiment is carried out with CNNs depth feature-based models to check the flexibility. VGG16 is a deep learning technique specially designed for image analysis and classification; it consists of 1-5 convolution and max-pooling layers with spades and filters. The depth is 21 layers. VGG16 Keras package is implemented in a python environment. VGG16 is a complex and time-consuming process in extracting features from colored frames. Figure 11 shows the performance comparison of each video type MAP, AUC, and Accuracy with all the methods.

The main reason behind the success of the proposed work is elimination of redundant frames using KPCA and the use of deep feature extraction using VGG16. In comparision, other such as HF [2] were built on hierarchical approach but could not manage redundant frames. Similarly, SePH [8] first uses particle component analysis(PCA) on the dataset and the spectral hashing on a binary bit of data but

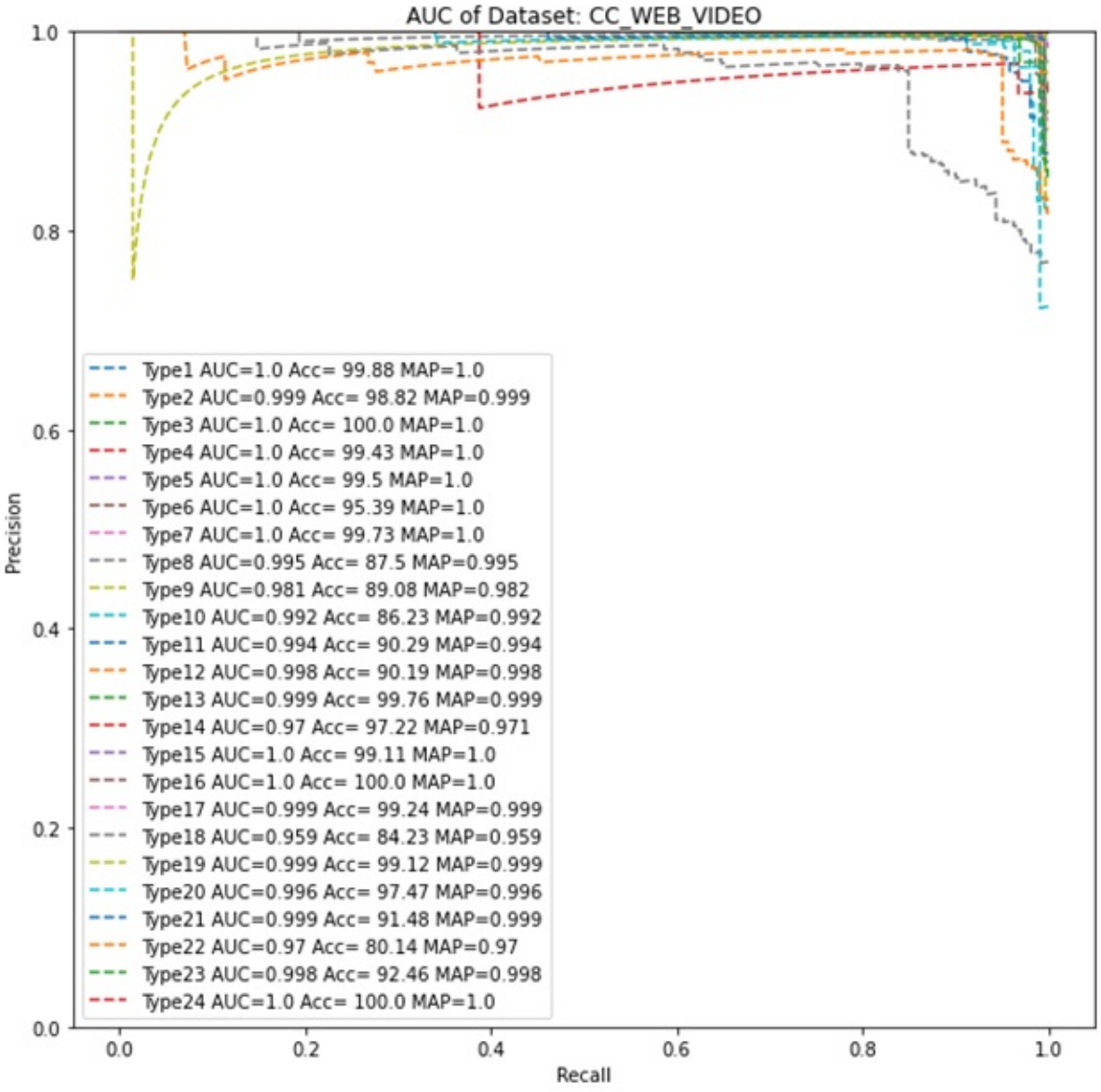

Figure 11. AUC, MAP, and Accuracy comparison using VGG 16 Algorithm

could not address the issue of non-linearity among the non-redundant frame. SPH [16] utilizes semantics preserving hashing techniques and uses a smaller number of data for training through kernel logistic regression with a sampling strategy. Thus, making the entire system more complex, and increases execution time and space. SMVH [18], MvDA [19] shows the improvement in the result relies on time-consuming distance computation and searching strategy and is not suitable for large-scale application.

ROC curve based on precision vs. recall is demonstrated in figure 8, which clearly shows the advantage of our approach on a large dataset. In figure 8, the average precision (AP), an area under curve, and accuracy are shown for all 24-video sets. In paper [32], precision, recall and MAP performance measures are discussed in detail.

C. *Efficiency Evaluation*

The volume of the dataset is the primary concern that is critically evaluating the performance of any NDVR on time and space complexity. Table V shows the average running time comparison among different methods to identify near-duplicate video from query video. Algorithm HF consumes extensive time as it does pair by comparison between query video and all the video databases. Thus, making it practically unfit for big data NDVR. SePH, SPH, and MFH consume a considerable amount of time in the training phase, but the algorithm is complex in searching real-

TABLE IV. Proposed Methods Comparison on Accuracy, Area Under Curve, and MAP on CC-WEB-VIDEO

Video	HSV			LBP			HSV-LBP			VGG16		
Type	Accuracy	AUC	MAP	Accuracy	AUC	MAP	Accuracy	AUC	MAP	Accuracy	AUC	MAP
VT1	95.2	0.9890	0.9890	99.14	0.999	0.999	95.200	0.989	0.989	99.880	1.000	1.000
VT2	97.14	0.9944	0.9944	97.98	0.999	0.999	97.140	0.994	0.994	98.820	0.999	0.999
VT3	95	0.9922	0.9923	96.14	0.990	0.990	95.000	0.992	0.992	100.000	1.000	1.000
VT4	96.32	0.9857	0.9858	98.87	0.995	0.995	96.320	0.986	0.986	99.430	1.000	1.000
VT5	98.27	0.9877	0.9877	97.03	0.998	0.998	98.270	0.988	0.988	99.500	1.000	1.000
VT6	87.84	0.9832	0.9833	75.29	0.993	0.993	87.840	0.983	0.983	95.390	1.000	1.000
VT7	88.22	0.9461	0.9464	78.42	0.936	0.936	88.220	0.946	0.946	99.730	1.000	1.000
VT8	97.1	0.9880	0.9880	62.75	0.986	0.986	97.100	0.988	0.988	87.500	0.995	0.995
VT9	98.74	0.9895	0.9902	73.11	0.994	0.994	98.740	0.990	0.990	89.080	0.981	0.982
VT10	92.13	0.9854	0.9854	73.44	0.957	0.957	92.130	0.985	0.985	86.230	0.992	0.992
VT11	55.38	0.9556	0.9557	83.2	0.974	0.974	55.380	0.956	0.956	90.290	0.994	0.994
VT12	83.39	0.9851	0.9851	93.09	0.993	0.993	83.390	0.985	0.985	90.190	0.998	0.998
VT13	97.37	0.9535	0.9544	95.7	0.872	0.876	97.370	0.954	0.954	99.760	0.999	0.999
VT14	96.3	0.9863	0.9865	86.11	0.918	0.919	96.300	0.986	0.987	97.220	0.970	0.971
VT15	94.25	0.9827	0.9827	90.62	0.980	0.980	94.250	0.983	0.983	99.110	1.000	1.000
VT16	89.9	0.9988	0.9988	68.75	0.994	0.994	89.900	0.999	0.999	100.000	1.000	1.000
VT17	90.87	0.9807	0.9807	94.82	0.986	0.986	90.870	0.981	0.981	99.240	0.999	0.999
VT18	78.24	0.9336	0.9352	80.84	0.954	0.955	78.240	0.934	0.935	84.230	0.959	0.959
VT19	86.84	0.9636	0.9637	87.02	0.930	0.930	86.840	0.964	0.964	99.120	0.999	0.999
VT20	90.4	0.9562	0.9574	84.34	0.923	0.924	90.400	0.956	0.957	97.470	0.996	0.996
VT21	93.23	0.9953	0.9953	78.82	0.995	0.995	93.230	0.995	0.995	91.480	0.999	0.999
VT22	81.31	0.9657	0.9657	85.28	0.970	0.970	81.310	0.966	0.966	80.140	0.970	0.970
VT23	86.32	0.9871	0.9872	82.17	0.982	0.982	86.320	0.987	0.987	92.460	0.998	0.998
VT24	99.36	1.0000	1.0000	84.66	0.997	0.997	99.360	1.000	1.000	100.000	1.000	1.000

time query videos. SMVH requires multiple iterations to build hash code, and a similar set has been reported in MvDA systems. The proposed system can generate single steps without iteration, thus saving time during the learning phase. Our approach produces a shorter-length hash code. These hash codes are used as a map for all frames, thus reducing the image volume for the process by considering unique hash codes. Later, using KPCA to reduce the size of final features further leads to increased efficiency in real-time analysis.

TABLE V. Time complexity comparison of methods on CC-WEB-VIDEO dataset

Systems	Time in 0.00001Sec
HF [2]	>8000
SePH [8]	8.134
SPH [16]	3.80
MFH [17]	3.94
SMVH [18]	5.84
MvDA [19]	3.71
MVH-HSV-SVM	2.25
MVH-LBP-SVM	2.02
MVH-HSV-LBP-SVM	3.5
VGG16-SVM	14.7

5. Conclusion

The significant challenges for any NDVR are accuracy and time and space complexity. Although most of the algorithms succeeded in gaining more than 90% accuracy, the cost of space and time complexity was reported to be higher. Maybe the dimension of the features extracted from frames is higher. This paper proposed a kernel PCA-based multi-View hashing to extract critical, meaningful features from HSV and LBP images to classify NDVR. Proposed methods simultaneously address achieving low space complexity, faster execution, and high accuracy. Each average video feature is considered to maximize local inter and global intra-label lookouts in the proposed system. The objective is to reserve the evidence from all features. Tough the kPCA-MVH is slightly low in accuracy and MAP compared with the VGG16 model, it is still much better in comparison w.r.t space and time complexity. In the future, we will try to test other deep learning models and tune its parameter to address space and time complexity on a similar type of dataset. We will also consider an effective fusion scheme by training end-to-end rather than acquiring pre-trained architecture.

References

[1] J. Liu, Z. Huang, H. Cai, H. T. Shen, C. W. Ngo, and W. Wang, "Near-duplicate video retrieval: Current research and future trends," *ACM Computing Surveys (CSUR)*, vol. 45, no. 4, pp. 1–23, 2013.

[2] X. Wu, A. G. Hauptmann, and C.-W. Ngo, "Practical elimination of near-duplicates from web video search," in *Proceedings of the 15th ACM international conference on Multimedia*, 2007, pp. 218–227.

[3] X. Nie, W. Jing, C. Cui, C. J. Zhang, L. Zhu, and Y. Yin, "Joint multi-view hashing for large-scale near-duplicate video retrieval," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 10, pp. 1951–1965, 2019.

[4] Y. Hao, T. Mu, R. Hong, M. Wang, N. An, and J. Y. Goulermas, "Stochastic multiview hashing for large-scale near-duplicate video retrieval," *IEEE Transactions on Multimedia*, vol. 19, no. 1, pp. 1–14, 2016.

[5] J. Song, Y. Yang, Z. Huang, H. T. Shen, and R. Hong, "Multiple feature hashing for real-time large scale near-duplicate video retrieval," in *Proceedings of the 19th ACM international conference on Multimedia*, 2011, pp. 423–432.

[6] M. Cherubini, R. De Oliveira, and N. Oliver, "Understanding near-duplicate videos: a user-centric approach," in *Proceedings of the 17th ACM international conference on Multimedia*, 2009, pp. 35–44.

[7] D. Zhang, J. Wang, D. Cai, and J. Lu, "Self-taught hashing for fast similarity search," in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, 2010, pp. 18–25.

[8] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," *Advances in neural information processing systems*, vol. 21, 2008.

[9] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 915–928, 2007.

[10] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[11] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[12] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 1999.

[13] W.-L. Zhao, C.-W. Ngo, H.-K. Tan, and X. Wu, "Near-duplicate keyframe identification with interest point matching and pattern learning," *IEEE Transactions on Multimedia*, vol. 9, no. 5, pp. 1037–1048, 2007.

[14] D. Xu, T. J. Cham, S. Yan, L. Duan, and S.-F. Chang, "Near duplicate identification with spatially aligned pyramid matching," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 8, pp. 1068–1079, 2010.

[15] Y. Ke and R. Sukthankar, "Pca-sift: A more distinctive representation for local image descriptors," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2. IEEE, 2004, pp. II–II.

[16] Z. Lin, G. Ding, M. Hu, and J. Wang, "Semantics-preserving hashing for cross-view retrieval," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3864–3872.

[17] J. Song, Y. Yang, Z. Huang, H. T. Shen, and J. Luo, "Effective multiple feature hashing for large-scale near-duplicate video retrieval," *IEEE Transactions on Multimedia*, vol. 15, no. 8, pp. 1997–2008, 2013.

[18] Y. Hao, T. Mu, R. Hong, M. Wang, N. An, and J. Y. Goulermas, "Stochastic multiview hashing for large-scale near-duplicate video retrieval," *IEEE Transactions on Multimedia*, vol. 19, no. 1, pp. 1–14, 2016.

[19] M. Kan, S. Shan, H. Zhang, S. Lao, and X. Chen, "Multi-view discriminant analysis," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 1, pp. 188–194, 2015.

[20] "Wikipedia-hsl and hsv," 2022, last accessed 2 March 2022, at 06:14 (UTC). [Online]. Available: https://en.wikipedia.org/wiki/HSL_and_HSV

[21] E. Patel and S. Krishnan, "Generating stylistic images by extending neural style transfer method," in *2020 3rd International Conference on Sensors, Signal and Image Processing*, 2020, pp. 19–24.

[22] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[23] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE international conference on computer vision*, vol. 2. IEEE, 1999, pp. 1150–1157.

[24] L. Shang, L. Yang, F. Wang, K.-P. Chan, and X.-S. Hua, "Real-time large scale near-duplicate web video retrieval," in *Proceedings of the 18th ACM international conference on Multimedia*, 2010, pp. 531–540.

[25] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 915–928, 2007.

[26] D. Huang, C. Shan, M. Ardabilian, Y. Wang, and L. Chen, "Local binary patterns and its application to facial image analysis: a survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 6, pp. 765–781, 2011.

[27] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[28] X. Zhu, J. Zhao, J. Yuan, and H. Xu, "A fuzzy quantization approach to image retrieval based on color and texture," in *Proceedings of the 3rd International Conference on Ubiquitous Information Management and Communication*, 2009, pp. 141–149.

[29] O. Barkan, J. Weill, L. Wolf, and H. Aronowitz, "Fast high dimensional vector multiplication face recognition," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 1960–1967.

[30] X. Wu, A. G. Hauptmann, and C.-W. Ngo, "Practical elimination of near-duplicates from web video search," in *Proceedings of the 15th ACM international conference on Multimedia*, 2007, pp. 218–227.

[31] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[32] D. A. Phalke and S. Jahirabadkar, "A survey on near duplicate video retrieval using deep learning techniques and framework," in *2020 IEEE Pune Section International Conference (PuneCon)*. IEEE, 2020, pp. 124–128.

Dhanashree Ajay Phalke received degree, BE Computer Science and Engineering from Dr. Babasaheb Ambedkar Marathwada Univeristy, Aurangabad, India in 2000 and ME Computer Engineering from Savitribai Phule Pune University (Formerly University of Pune), India in 2008. She is pursuing PhD degree in Computer and Information Technology at Department of Technology, Savitribai Phule Pune University, India. Since 2001, she is working as an Assistant Professor in D. Y. Patil College of Engineering, Akurdi, Pune, India. She has published 50+ research papers in various journals and conferences.

Sunita Jahirabadkar is working as Dean (R&D) and Professor in the Department of Computer Engineering, Cummins College of Engineering for Women, Pune from last 23 years. She has nearly 27 years of teaching, industry and research experience. She has more than two dozens research publications in various International conferences and journals. She has worked on many funded research projects. Currently she is working on a very prestigious research project in collaboration with ISRO. She has served as a reviewer as well as session chair, for many of the International Conferences/Journals in her areas of expertise. She is the author of two books "e-business" published by Oxford University Press and "Big Data Analytics" by PHI Learning. Apart from this, she has written few book chapters as co-author. Her areas of interests include Business Intelligence, Machine learning, Data Analytics, Distributed systems, Artificial Intelligence etc. Her recent area of interest (or area of studies) includes "IPR" i.e. Intellectual property Rights. She has delivered few lectures on "IPR" and written articles for magazines on the same. Recently Dr. Sunita has received "Women in Science and Innovation Award" by the "Global Education and Corporate Leadership (GECL-2020) Awards.